Review: Planning

- Specialized methods for a particular kind of task:
  - Representation and algorithms less general but more efficient
  - Note that even human brain appears to have specialized hardware for many tasks
    - Vision, speech, …
Planning

• Given: start and goal states, and a set of operations
• Find: a sequence of operations that gets from start to goal
• What is special about this problem?
  – State is a set of assertions (facts, FOL sentences)
  – Operations change a subset of the assertions
  – Limited interaction between subproblems

The Strips Language for Planning

• states are represented as conjunctions of function-free ground literals, e.g.,
  At(Home) ^ ~ Have(Milk) ^ ~Have(Bananas) ^ …
  – states may be incomplete - positive literals not mentioned are assumed to be false.
• goals are conjunctions of literals, and can contain variables.
  – Literals not mentioned are assumed “don’t care”
Representing Actions in Strips

- action description
  - Possibly parameterized, e.g. go(There)
- Precondition, e.g. at(Here)
- Effect
  - Add list, e.g. at(There)
  - Delete list, e.g. at(Here)

Representing Plans

- A plan is a graph whose nodes are actions
  - Also dummy actions for initial & final states
- Arc to each precondition showing which previous action makes that precondition true

\[
\text{Go(grocery store)}
\]

\[
\text{At(grocery store)}
\]

\[
\text{At(grocery store)}
\]

\[
\text{Buy(milk)}
\]
Representing Plans

- Also precedence arcs
  - This action must go before that action
  - Added to protect a precondition from being undone between time established and time needed.

Building Plans

- Progression
- Means-Ends
- Regression
Progression

Plan(S,G):
If (S satisfies G)
then Return(NIL)
else a. Let A be the set of operator instantiations applicable to S.
b. If A = then Fail.
c. Let I = Choose from A.
d. Return(Cons(I,Plan(Apply(I,S),G))).

Evaluation of Progression

• Not bad for small number of operators.
• Branching factor grows as we add operators.
  – E.g., Operators for painting blocks.
  – E.g., Operators for rotating blocks.
• Branching factor grows as we add objects.
  – E.g., More blocks.
  – E.g., More paint colors.
• Operators are chosen without regard to their relevance to achieving goals.
• Algorithm is complete.
Means-Ends Planning

\[
\text{Plan}(S, G):
\text{If (S satisfies G)} \\
\text{then Return(NIL)}
\]
\text{else a. Let A = Set of operator instantiations that add a} \\
\text{literal L that is in G and is not satisfied in S.} \\
\text{b. If A = \{\} then Fail.} \\
\text{c. Let I = Choose from A.} \\
\text{d. Let P = Plan(S, Preconditions(I)).} \\
\text{e. Let S = Apply(P, S).} \\
\text{f. Let S = Apply(I, S).} \\
\text{g. Let R = Plan(S, G).} \\
\text{h. Return(Append(P, Cons(I, R))).}
Evaluation of Means-Ends Planning

– Operators are chosen with an eye toward their relevance to achieving goals.
– Algorithm is not complete.

Regression (Backward Chaining)
Regression (Goal stack planning)

Plan(S,G):
If (S satisfies G) then Return(NIL).
else a. Let A = Set of operator instantiations that add a literal L in G, and that delete no literal in G.
b. If A = {} then Fail.
c. Let I = Choose from A.
d. Let G = Regress(G,I).
e. Let P = Plan(S,G).
f. Return(Append(P,List(I)).

Goal Regression

• apply an action backwards to produce subgoal expressions.
  – unify one of the literals in the (sub)goal expression with one of the literals in the add list of the rule.
  – subgoal expression is created by regressing the other (nonmatched) literals in the goal expression through the instantiated rule,
  – and conjoining these with the preconditions of the instantiated rule.
Examples

- **Goal:** \([ON(A,B) \land ON(B,C)]\), **Action:** stack\((A,B)\)
  - ON\((A,B)\) unifies with ON\((X,Y)\) effect of stack, with \(A/X, B/y\)
  - Regress rest of goal, i.e., ON\((B,C)\) through stack\((A,B)\), giving ON\((B,C)\).
  - Add preconditions HOLDING\((A)\), CLEAR\((B)\)
  - to yield the subgoal:
  - \([ON(B,C) \land HOLDING(A) \land CLEAR(B)]\)

- **Goal:** CLEAR\((A)\), **Action:** unstack\((x,A)\)
  - Unifies with CLEAR\((y)\) effect of unstack\((x,y)\)
  - Add preconditions to get subgoal \([HANDEMPTEY \land CLEAR(x) \land ON(x,A)]\).

Examples

- **Goal:** \([CLEAR(A) \land HANDEMPTEY]\), **Action:** unstack\((x,A)\)
  - Unifies with CLEAR\((y)\) effect of unstack\((x,y)\)
  - But when we try to regress HANDEMPTEY through clear\((A)\) we fail, because HANDEMPTEY is delete list of clear.
Regression on Sussman Anomaly

GOAL DESCRIPTION: ON(A,B)* ON(B,C)
INITIAL STATE: ON(C,A) ONTABLE(A)
ONTABLE(B) CLEAR(B)
CLEAR(C) HANDEMPY
REGRESS STACK(A,B): HOLDING(A)* CLEAR(B)

REGRESS PICKUP(A): ONTABLE(A) CLEAR(A)
HANDEMPY*

REGRESS STACK(B,C): HOLDING(B)* CLEAR(C)

REGRESS PICKUP(B): ONTABLE(B) CLEAR(B)
HANDEMPY*

Regression on Sussman Anomaly: Cont'd

REGRESS STACK(B,C): HOLDING(B)* CLEAR(C)

ONTABLE(A) CLEAR(A)

REGRESS PICKUP(B): ONTABLE(B) CLEAR(B)
HANDEMPY*

CLEAR(C)* ONTABLE(A)
CLEAR(A)
Regression on Sussman
Anomaly: Cont'd

REGRESS PUTDOWN(C): HOLDING(C)*

..............................
ONTABLE(B) CLEAR(B)
ONTABLE(A) CLEAR(A)*

REGRESS UNSTACK(C,A): ON(C,A)
CLEAR(C)

HANDEMPTY

..............................
ONTABLE(B) CLEAR(B)
ONTABLE(A)

Evaluation of Regression

– Operators are chosen with an eye toward
– their relevance to achieving goals.
– Algorithm is complete.
Abstraction

Problem -> Abstract Problem
--> Abstract Solution -->Concrete Solution
Example: hierarchical planning
Causes of Plan Failure

- bounded indeterminacy: actions can have unexpected results, but the possible effects can be enumerated and built into the action description axiom.
- unbounded indeterminacy: the set of possible outcomes is too large to enumerate. We must replan when our predictions are wrong.